SCALABILITY AND PERFORMANCE ANALYSIS OF EC-EARTH 3.2.0 USING A NEW METRIC APPROACH (PART I)

BSC-CES-2016-001

EC-Earth 3.2.0, Scalability, Metrics, Computational Performance

X. Yepes-Arbós, M. C. Acosta, K. Serradell, O. Mula-Valls, F. J. Doblas-Reyes

Earth Sciences Department

Barcelona Supercomputing Center - Centro Nacional de Supercomputación (BSC-CNS)

20 July 2016

TECHNICAL REPORT

Series: Earth Sciences (ES) Technical Report

A full list of ES Publications can be found on our website under:

https://earth.bsc.es/wiki/doku.php?id=library:external:technical_memoranda

Barcelona Supercomputing Center–Centro Nacional de Supercomputación (BSC-CNS)

C/Jordi Girona, 31 | 08034 Barcelona (Spain)

Summary

In this technical memorandum we present a complete scalability analysis to determine the computational performance of the EC-Earth 3.2.0 climate model. Climate models like EC-Earth usually consume a very large quantity of resources at the same time as results being expected in a reasonable execution time. This is even more important when a limited number of hours is allocated on a specific high performance platform. For this reason, a correct configuration avoids the loss of resources.

After performing the scalability analysis, we are able not only to identify some MPI combinations, but also to propose understandable metrics for users to choose the best configuration for their specific needs. The experiments use the T255L91 grid for IFS and the ORCA1L75 grid for NEMO. They were executed on MareNostrum III, hosted by the Barcelona Supercomputing Center. This text describes the configuration with the best speedup of EC-Earth, which is 40.3 with 15.7 SYPD, using 640 MPI processes distributed between 512 for IFS and 128 for NEMO. In contrast, the combination that uses 416 MPI processes, dedicating 288 to IFS and 128 to NEMO, has the best performance-efficiency compromise, achieving a speedup of 35 and 13.6 SYPD, but using less resources.

The document also describes how to estimate the number of MPI processes to achieve a desired throughput. This allows an estimation of the number of MPI processes that achieves a given value of simulated years per day, taking into account a good compromise between performance and efficiency.

Contents

1.Introduction

Recently, there have been great improvements in the quality of climate models. One of the reasons is that computational power has grown exponentially. This leads to an enormous complexity of the models and of the experiments that could be carried out. However, it is still uncommon to find the widespread use of metrics that evaluate the performance of a model.

From a computational point of view, there is available the speedup metric, which evaluates the improvement in speed of execution time of a parallel application using different amount of resources. Although speedup is one of the most used metrics in computer science, it could not satisfy the scientists' needs. Scientists are interested in the throughput and the scalability of their models, because of the need to simulate a desired time period within a specific time to solution. For this reason, in this study we present other metrics such as the simulated years per day (SYPD) metric, which tells about the throughput of a climate model on a supercomputer, in the case of this document EC-Earth 3.2.0 on MareNostrum III, using different MPI combinations.

EC-Earth is a global coupled climate model integrating a number of component models to simulate the Earth system. The two main models are IFS 36r4 as atmospheric model and NEMO 3.6 as ocean model, both coupled using OASIS3-MCT. There are other small components like LIM3 as sea-ice model and runoff-mapper to distribute runoff from land to the ocean through rivers.

Scientists also have other concerns, such as to consume the minimum resources to perform the simulation. This requires performing slower rather than just one fast simulation. This is related to the efficiency of the model, but also to the speedup. As a consequence, a compromise between performance and efficiency is needed, which should be applied to the MPI combinations of the SYPD results. This is the idea developed in this technical memorandum.

The analysis used has been developed taking into account that the scalability analysis is bi-dimensional, setting the MPI processes used for the two model components, NEMO and IFS, at the same time. More details about our definition of a bi-dimensional search can be found in section 3.1.

Although other scalability analyses [1] have been performed in the past, one of the main goals of this document is to try to improve the information made available up to now to scientists. For example, one of the weaknesses of other studies is that the scalability results are hard to understand and show partial information. They use few MPI combinations, not representative enough, especially when considering that some combinations could be hidden in a bi-dimensional search. Furthermore, they present the quantity of IFS and NEMO processes together, making difficult to distinguish the combination of cores used for NEMO and IFS independently. On the other hand, efficiency results are not realistic as they normalize data with respect to a combination that has many processes. The base case of the efficiency plot always has value 1, which means that the number of processes used for this base case is 100% efficient, which in not always true. For a sequential program (1 process), this is always true, but using 2 or more processes, this is probably false, which is the case of EC-Earth.

Thus, if the scalability analysis of EC-Earth starts with a base case using many processes, in order to plot a value of 1 in the base case of the efficiency, all the data has to be normalized regarding the number of processes of that base case, which leads to have better efficiencies, but it is unrealistic.

The document is organized as follows. The experimental setup, including model description, environment and model configuration, is given in the next section. The methodology used is described in section 3. Section 4 explains the results of the scalability analysis. Finally, section 5 contains the conclusions of the whole study.

However, in the future we will release a new version of this document adding a complete performance analysis of EC-Earth 3.2.0 using HPC tools. It is necessary to find out the main bottlenecks of the model that are limiting its throughput. It will emphasize the coupling between IFS and NEMO because having coupled models represents a big challenge in terms of good performance and efficiency. Furthermore, it will be necessary to re-perform a scalability analysis for the high configuration that EC-Earth will have available in the near future.

2. Experimental setup

2.1. Model description

EC-Earth [2] is a global coupled climate model, which integrates a number of component models in order to simulate the earth system. It is developed by a consortium of European research institutions, which collaborate in the development of a new Earth System Model. The goal of EC-Earth is to build a fully coupled atmosphere-ocean-land-biosphere model usable for problems encompassing from seasonal-to-decadal climate prediction to climate change projections and paleoclimate simulations. It includes the following components:

- The **OASIS3-MCT coupler**: is a coupling library to be linked to the component models and which main function is to interpolate and exchange the coupling fields between them to form a coupled system.

- The **Integrated Forecasting System (IFS)** as atmosphere model: is an operational global meteorological forecasting model developed and maintained by the European Centre of Medium-Range Weather Forecasts (ECMWF). The dynamical core of IFS is hydrostatic, two-time-level, semi-implicit, semi-Lagrangian and applies spectral transforms between grid-point space and spectral space. In the vertical the model is discretised using a finite-element scheme. A reduced Gaussian grid is used in the horizontal. The IFS cycle is 36r4.

- The **Nucleus for European Modelling of the Ocean (NEMO)** as ocean model: is a state-of-the-art modelling framework for oceanographic research, operational oceanography seasonal forecast and climate studies. It discretizes the 3D Navier-Stokes equations, being a finite difference, hydrostatic, primitive equation model, with a free sea surface and a non-linear equation of state in the Jackett. The ocean general circulation model (OGCM) is OPA (Océan Parallélisé). OPA is a primitive equation model which is numerically solved in a global ocean curvilinear grid known as ORCA.

  EC-Earth 3.2.0 uses NEMO's version 3.6 with **XML Input Output Server (XIOS)** version 1.0. XIOS is an asynchronous input/output server used to minimize previous I/O problems.

- The **Louvain-la-Neuve sea-Ice Model 2/3 (LIM2/3)**: is a thermodynamic-dynamic sea-ice model directly coupled with OPA.

- The **Hydrological extension of the Tiled ECMWF Surface Scheme for Exchange processes over Land (HTESSEL)** as land and vegetation module: is part of the atmosphere model. It solves the surface energy and water balance taking into account 6 different land tiles overlying a 4 layer soil scheme. The 6 tiles are: tall vegetation, low vegetation, interception reservoir, bare soil, snow on low vegetation, and snow under high vegetation.

- The **Tracer Model 5 (TM5)** as global chemistry transport model: describes the atmospheric chemistry and transport of reactive or inert tracers.

- The **runoff-mapper** component is used to distribute the runoff from land to the ocean through rivers. It runs using its own binary and coupled through OASIS3-MCT.

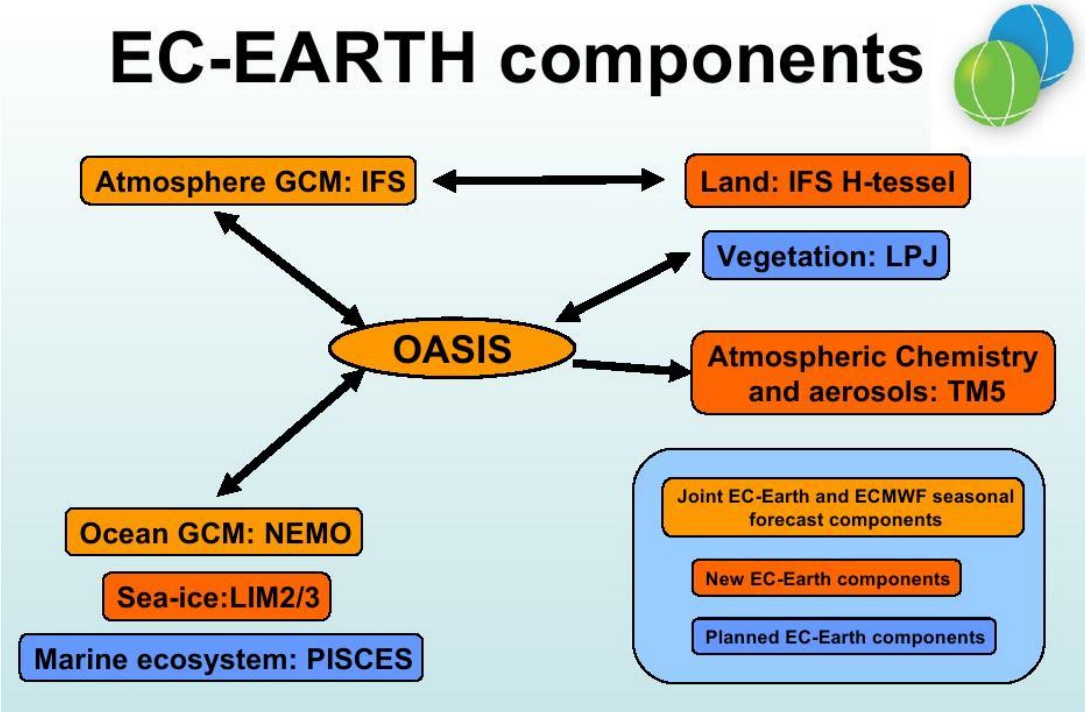

Figure 1: Components used for the EC-Earth model

The components are mainly written in Fortran, except XIOS which is written in C++. The model is parallelized using the message passing paradigm, particularly, the standard library MPI (Intel MPI v5) to be executed in distributed HPC clusters with several nodes.

For our performance analysis, we will use IFS, NEMO, LIM3, XIOS and OASIS, being these components the essential part of EC-Earth and the modules mainly used for BSC scientists in the experiments.

2.2. Environment

All analysis were done on MareNostrum III, which is a supercomputer designed by IBM and it is hosted by the Barcelona Supercomputing Center (BSC). It has the next features:

- Peak performance of 1.1 PFLOPS
- 115.5 TB of main memory
- 2 PB of disk storage
- Main nodes:
    - 2x Intel SandyBridge-EP E5-2670/1600 20M 8-core at 2.6 Ghz
    - 128 nodes with 16x8 GB DDR3-1600 DIMMS (8 GB/core)

  - 128 nodes with 16x4 GB DDR3-1600 DIMMS (4 GB/core)
  - 2752 nodes with 8x4 GB DDR3-1600 DIMMS (2 GB/core)
  - 500 GB 7200 rpm SATA II local HDD
- Operating system: Linux - SuSe distribution
- LSF queue system
- Interconnection networks:
  - Infiniband FDR10
  - Gigabit Ethernet
  - 52 racks distributed in 120 m²

2.3. Model configuration

In order to perform the analysis explained in section 3, it is necessary to set the configuration, which cannot be changed because we want to compare performance analysis. We will run the last version available, the coupled EC-Earth 3.2.0 model, so we have the following conditions:

- Revision: 2716. 2015-12-08 15:14:02 +0100 (Tue, 08 Dec 2015)
- MareNostrum III with 16 MPI processes per node. We use generic nodes with 32 GB and 16 cores, so we fill them up. The communications are done via Infiniband and the I/O is done via a Gigabit Ethernet network.
- 1 chunk of 3 months (to minimize initialization's overhead)
- Average of 3 identical executions
- Without check flags
- Default namelists values
- Optimum mpirun order: in order to exploit better processor affinity. Order: IFS, NEMO, Runoff-mapper and XIOS.
- Use of an optimization to avoid mpi_allgather use a the northfold
- Fortran compilation flags: -O2 -r8 –xHost
- C compilation flags: -O2 –xHost
- Compiler: Intel v13.0.1
- Intel MPI library v5.0.1.035
- MKL library v11.1.2
- NetCDF library v4.3.2
- HDF5 library v1.8.12-mpi
- GRIB library v.1.14.0

Furthermore, we have the following specific parameters:

Model	Nemo-3.6
Configuration name	ORCA1L75_LIM3
Resolution	ORCA1L75
Modules	OPA and LIM3
Time step	2700 seconds (32 time steps per day)
Compilation keys	key_trabbl key_vvl key_dynspg_ts key_ldfslp key_traldf_c2d key_traldf_eiv key_dynldf_c3d key_zdfddm key_zdftmx key_mpp_mpi key_zdftke key_lim3 key_iomput key_oasis3 key_oa3mct_v3

Table 1: Configuration parameters for NEMO

Model	ifs-36r4
Resolution	T255L91
Time step	2700 seconds (32 time steps per day)

Table 2: Configuration parameters for IFS

Component model	OASIS3-MCT
Coupling frequency	2700 seconds (default value)

Table 3: Configuration parameters for OASIS3-MCT

On the other hand, we used Autosubmit to automate the scalability analysis. It is a python-based tool to create, manage and monitor experiments by using computing clusters, HPCs and supercomputers remotely via ssh. It has support for experiments running in more than one HPC and for different workflow configurations.

3. Methodology

3.1. Scalability

We do the scalability analysis with the following goals:

- How EC-Earth 3.2.0 scales, and thus, its performance.

- Which are the ideal combinations of MPI processes taking into account different points of view, i.e., find out which is the fastest combination, or also find out a combination with a good performance-efficiency compromise according to our requirements. In order to do this, the results present not only a computational and scientific perspective, but also a new one, achieving a compromise between both of them, easy to understand by anyone.

- How many MPI processes have to be set to achieve a desired throughput. This means that the reader will be able to easily know the number of MPI processes to use if he or she wants to achieve a particular value for speedup, efficiency or simulated years per day. It will be also possible to tune this combination taking into account the performance-efficiency compromise, i.e., if the reader prefers more performance or more efficiency.

It is important to mention that the scalability analysis of EC-Earth might be one-dimensional, i.e., obtaining the best number of processes for IFS and NEMO independently. This is because it is not clear if IFS and NEMO models are dependent between them in terms of execution time. This means that if the number of processes of one model is fixed and we only iterate over the number of processes of the other model, the execution time of the model with fixed processes should not be affected and have always the same value. Coupling profiling tools such as LUCIA will be used for successive tests in order to clarify this and determine if IFS and NEMO models are dependent or not.

We present a methodology that is suitable for scalability analysis of dependent models (bi-dimensional search), but it is still valid for independent models (one-dimensional search). The difference between a one-dimensional and a bi-dimensional analysis is the amount of tests to be performed, being larger in the bi-dimensional.

However, taking into account the third goal presented above, in this study is necessary to use the bi-dimensional search, as we want to make easy the interpretation of the speedup, efficiency and simulated years per day charts. Otherwise, using the one-dimensional search would not be possible.

So using a bi-dimensional search, we programmed a script that handles all the work of the scalability analysis. Figure 2 shows the flowchart that follows our script. This script sets different number of NEMO and IFS processes for several MPI combinations automatically. Besides IFS and NEMO processes, we also have to set one process for XIOS and one process for

Runoff-mapper.

The criterion used to choose the MPI combinations of IFS and NEMO processes is based in powers of two: 1, 2, 4, 8, 16, 32, 64, 128, 256 and 512 for both IFS and NEMO, except 1, which is used only by IFS. NEMO is executed from 2 processes because we were not able to run it in coupled mode with 1 MPI process for these tests.

However, this is an exponential growth, so the distance between numbers becomes too large. These holes are filled up with multiples of two: 192, 224, 288, 320 and 384 for both IFS and NEMO, and 640, 768 and 896 only for IFS.

For each iteration of the MPI combinations, we create a new experiment with three identical members that are used to get the average time of three identical executions. Furthermore, we store each experiment ID in a file, which is used in the post-processing script. But we can also use the IDs' file to re-perform the scalability analysis in a future without creating all the experiments again.

The post-processing script reads the three execution times of each MPI combination, does the average and stores it again.

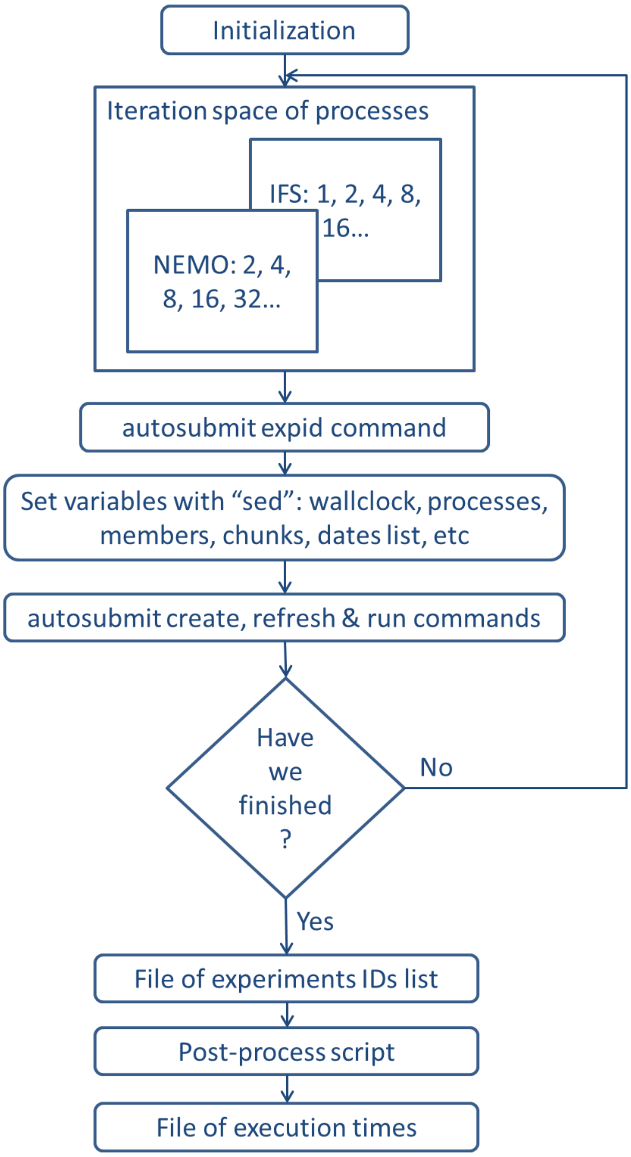
Initialization
Iteration space of processes
IFS: 1, 2, 4, 8, 16…
NEMO: 2, 4, 8, 16, 32…
autosubmit expid command
Set variables with "sed": wallclock, processes, members, chunks, dates list, etc
autosubmit create, refresh & run commands
Have we finished ?
No
Yes
File of experiments IDs list
Post-process script
File of execution times

Figure 2: Flowchart of the scalability analysis methodology

Using the execution time and the number of MPI processes, we can calculate the speedup, the efficiency and the simulated years per day. The formulas are:

$$Speedup = \frac{Execution\ time\ base\ combination}{Execution\ time}$$

Where:

- $Execution\ time\ base\ combination$ is the execution time of the combination that uses fewer MPI processes.
- $Execution\ time$ is the execution time of the rest of combinations that use more MPI processes than the base combination.

$$Efficiency = \frac{Speedup}{\left(\frac{\#MPI\ processes}{\#MPI\ processes\ base\ combination}\right)}$$

Where:

- $\#MPI\ processes\ base\ combination$ is the number of MPI processes of the combination that uses fewer MPI processes. It is used to normalize the data when the base case uses 2 or more processes, which is the case of EC-Earth.
- $\#MPI\ processes$ is the number of MPI processes used by a combination to get its $Speedup$.

$$SYPD = \left(\frac{3\ \cancel{months}}{Execution\ time\ \cancel{(s)}}\right)\left(\frac{1\ year}{12\ \cancel{months}}\right)\left(\frac{3600\ \cancel{s}}{1\ \cancel{h}}\right)\left(\frac{24\ \cancel{h}}{1\ day}\right) = \frac{21600}{Execution\ time\ (s)}\ {}^{Years}/_{Day}$$

Where:

- $Execution\ time\ (s)$ is the execution time of a combination expressed in seconds.

Since the scalability analysis is bi-dimensional and one of the goals is to provide an easy and understandable representation of the data, we have to define a process of selecting it accurately.

To do so, we decided to reduce the amount of data to show by selecting the most relevant and important one. Our methodology was to introduce a new metric, which gives a value to each MPI combination. This metric evaluates the performance-efficiency compromise in order to represent and study only the most relevant IFS-NEMO combinations. It is calculated as follows:

$$Performance\text{-}Efficiency\ Compromise = Speedup \times Efficiency$$

This formula penalizes combinations that have a low efficiency, but benefits the ones that have a good efficiency. It is used to get a two dimensional mask that we apply later to the common metrics: Execution time, speedup, efficiency and simulated years per day. With this mask we can obtain 2D charts of the listed metrics.

The procedure used to get this mask is based on creating subsets of MPI processes, first for IFS and then for NEMO. This means to fix IFS processes and iterate over all NEMO processes. To clarify it, it is done as follows:

- Subset 1.1: IFS = 1 and NEMO = 2, 4, 8, 16…
- Subset 1.2: IFS = 2 and NEMO = 2, 4, 8, 16…
- Subset 1.3: IFS = 4 and NEMO = 2, 4, 8, 16…
- Subset 1.4: IFS = 8 and NEMO = 2, 4, 8, 16…
- …

Then, for each 1.x subset we take the best value. We do the same process by fixing NEMO processes and iterating over all IFS processes:

- Subset 2.1: NEMO = 2 and IFS = 1, 2, 4, 8…
- Subset 2.2: NEMO = 4 and IFS = 1, 2, 4, 8…
- Subset 2.3: NEMO = 8 and IFS = 1, 2, 4, 8…
- Subset 2.4: NEMO = 16 and IFS = 1, 2, 4, 8…
- …

Again, we take the best value for each 2.x subset. This gives a mask of MPI combinations that have the best performance-efficiency compromise. The mask has combinations over all processes range, i.e., from 1 to 896 in IFS and 2 to 512 in NEMO. Otherwise, if we had not used the subset methodology, we only would have got a subrange of combinations (the ones with higher values).

On the other hand, once we have the mask, we have to apply it into the common metrics that we listed above. Using this, we get a list with the relevant data instead of the original table with all the data. Then, we sort this list by ascending order of total MPI processes (IFS + NEMO). At this point, we are able to represent the data in a 2 dimensional chart.

4. Results

4.1. Scalability

In the first part, we only present all the data with 3D charts. Then, we go into details with 3 subsections that analyze the results more accurately.

Note that the 3D charts are rotated to maximize the visualization of the data, so IFS and NEMO processes axes are not always in the same position and/or direction. Figures 3, 4, 5, 6 and 7 show the execution time, speedup, efficiency, simulated years per day and performance-efficiency compromise respectively.

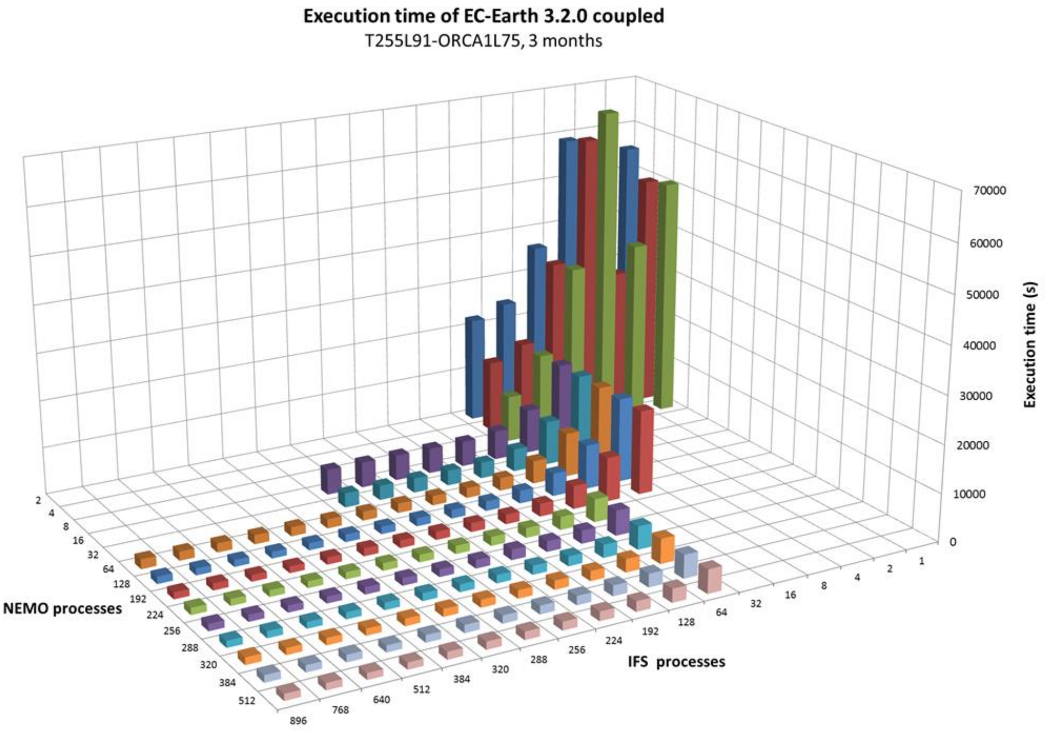

Figure 3: Execution time of EC-Earth 3.2.0 coupled in a 3D chart

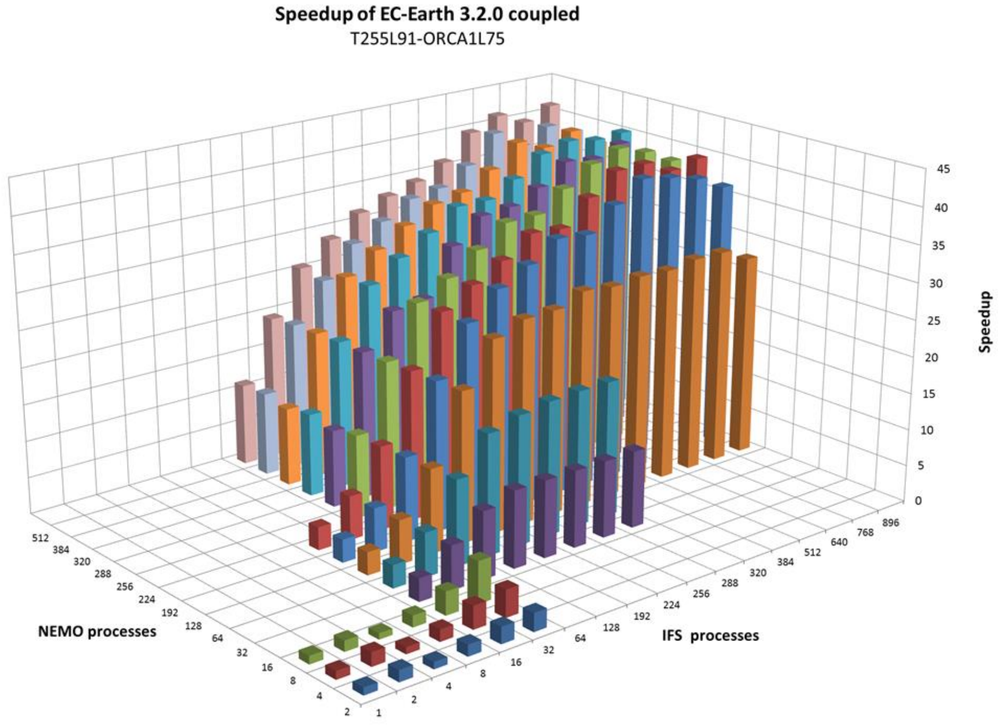

Figure 4: Speedup of EC-Earth 3.2.0 coupled in a 3D chart

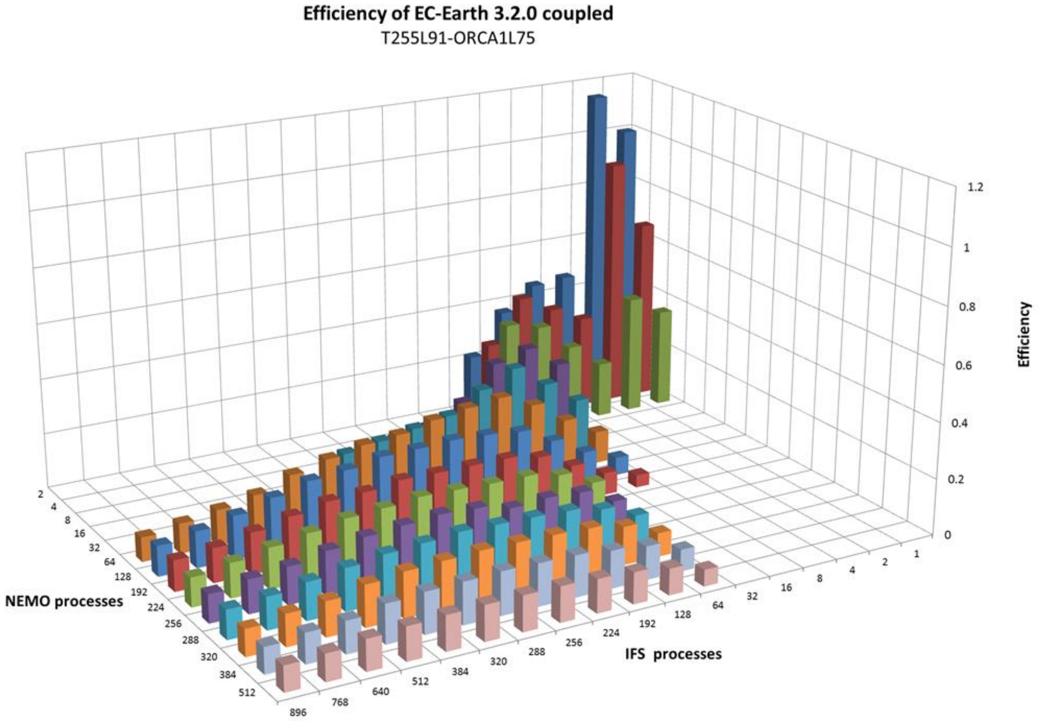

Figure 5: Efficiency of EC-Earth 3.2.0 coupled in a 3D chart

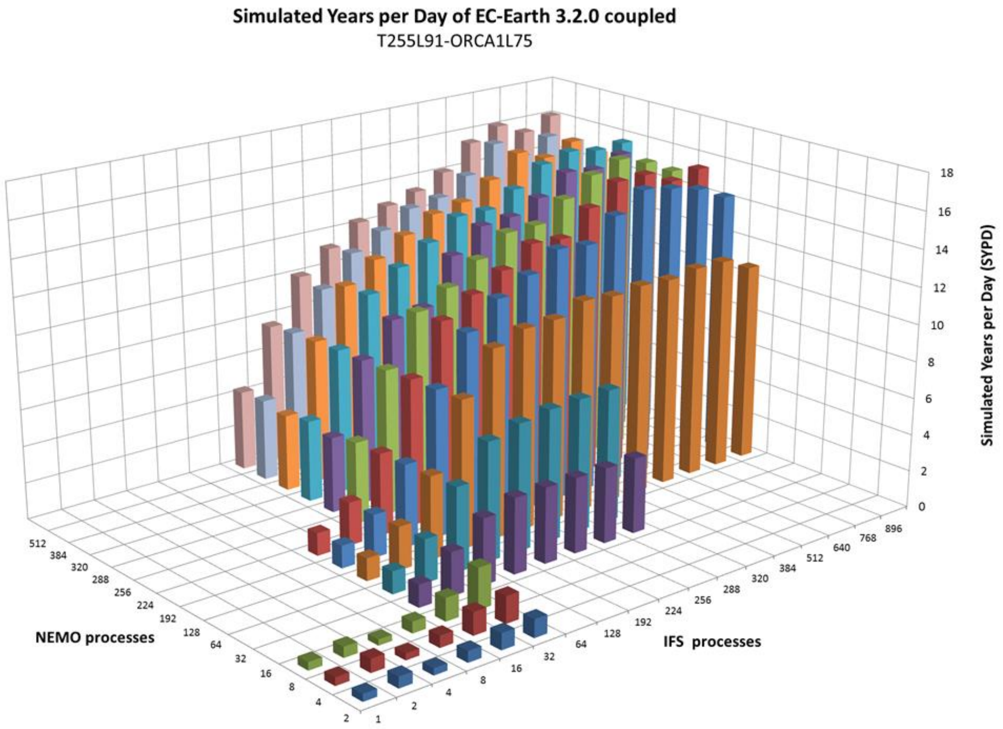

Figure 6: Simulated Years per Day of EC-Earth 3.2.0 coupled in a 3D chart

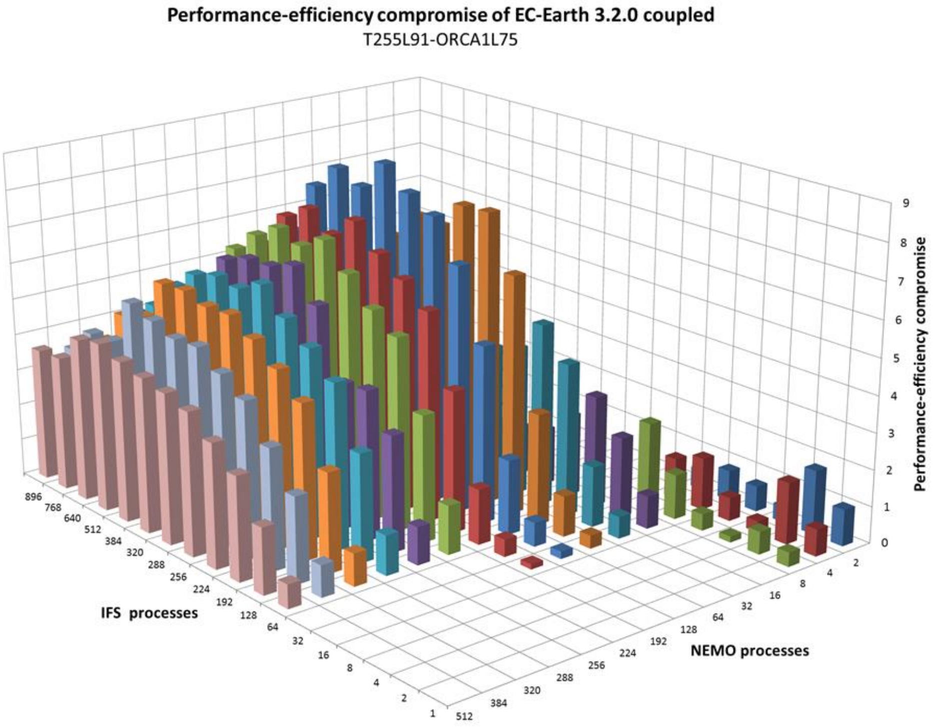

Figure 7: Performance-efficiency compromise of EC-Earth 3.2.0 coupled in a 3D chart

These charts show that EC-Earth 3.2.0 does not scale well, because the efficiency is really low. But clearly, interpreting these 3D charts is hard, so the following charts are in 2D to make the analysis easier.

4.1.1. Computational perspective

In this subsection we will present how EC-Earth 3.2.0 scales and we will identify which is the fastest combination and the more efficient. Note that to express the number of processes, we use this notation: "Total number of processes [IFS+NEMO]", where the two values within the brackets express how the "Total number of processes" are distributed between IFS and NEMO.

Applying the methodology explained in section 3.1, we make the charts of Figures 8, 9 and 10 which show execution time, speedup and efficiency respectively. We can see again and more clearly that EC-Earth 3.2.0 does not scale well at all. The efficiency is truly bad, as it plummets. There is an exception with the 4 [2+2] combination, which gives a super-linear speedup (efficiency above 1), but it is not useful because the speedup is too small. This super-linear speedup could be related to memory issues, like available bandwidth per process, amount of memory available per process, memory accesses collisions, etc.

The reason why EC-Earth 3.2.0 does not scale well is because using more processes implies higher fine-grained decomposition and higher overhead of the MPI communications, and this leads to important imbalances. This also explains the low efficiency.

Despite the bad performance, the speedup chart also shows that it increases relatively constant up to 416 [288+128] processes and, at most, up to 640 [512+128] processes. At this point, it becomes more or less flat.

On the other hand, the fastest combinations are 640 [512+128] and 896[768+128], but taking into account the number of processes, and as a consequence the efficiency, it is only worth to point out the 640 [512+128] combination as the fastest.

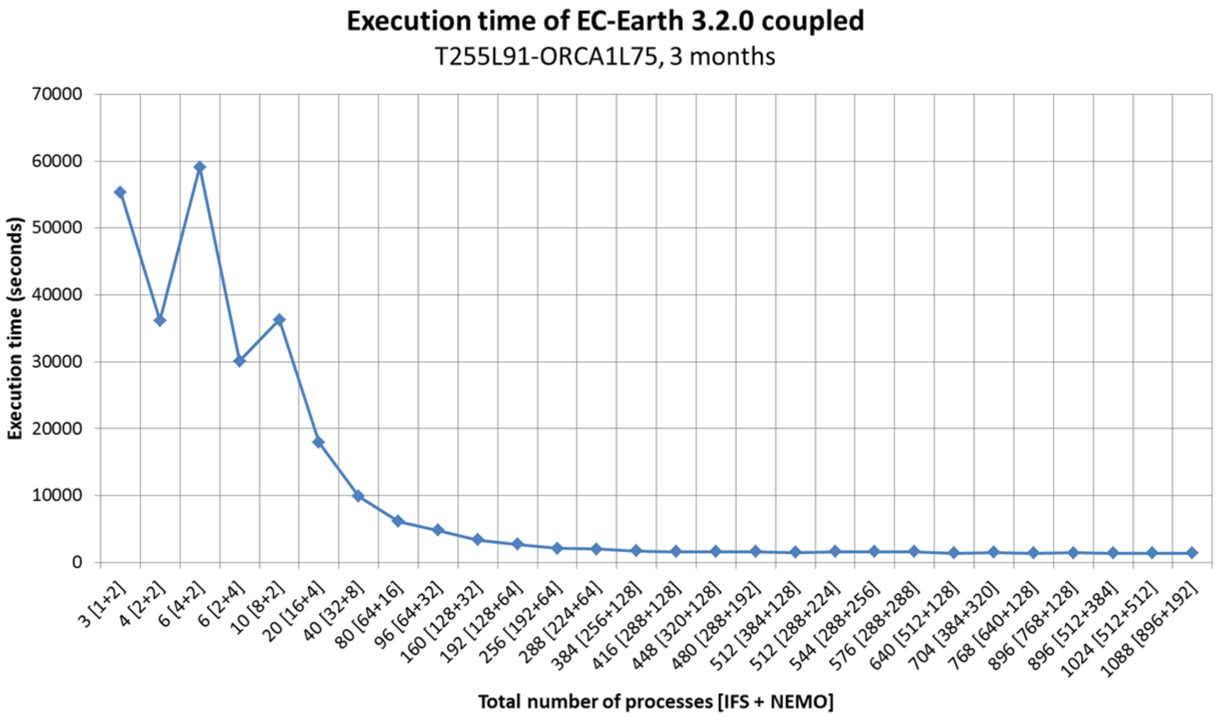

Figure 8: Execution time of EC-Earth 3.2.0 coupled in a 2D chart

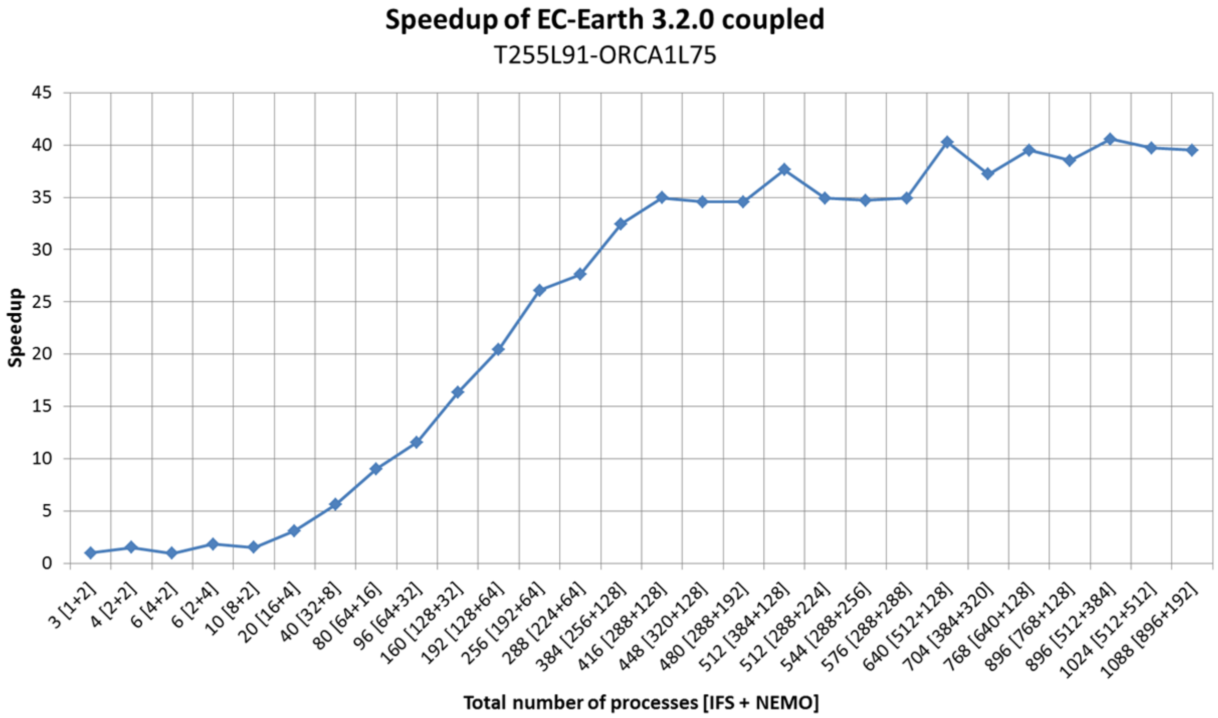

Figure 9: Speedup of EC-Earth 3.2.0 coupled in a 2D chart

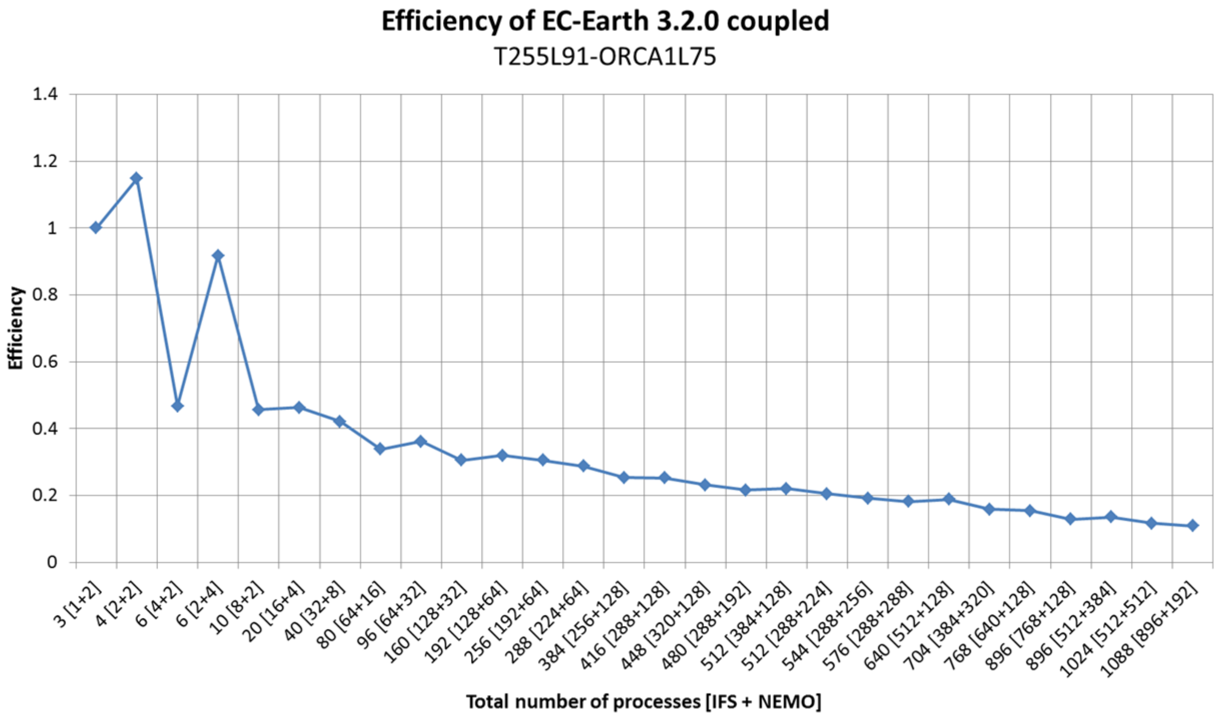

Figure 10: Efficiency of EC-Earth 3.2.0 coupled in a 2D chart

4.1.2. Scientific perspective

It is useful to choose the amount of MPI processes needed to achieve a desired throughput. For this purpose, the scientific community uses the simulated years per day metric, which tells you how many years can be simulated in a day of real life. Figure 11 shows this metric. Ideally, to perform an execution at a desired rate, one would select the SYPD on the vertical axis and read off the MPI combination that achieves it.

The speedup and SYPD charts have the same shape. This is because they are in function of the execution time and a constant value. As we saw in Figure 9, there is a steady increase up to 416 [288+128] processes in the SYPD chart, and, at most, we can get about 15.7 SYPD with 640 [512+128] and 896 [768+128] combinations. But taking into account the efficiency, it is only worth to use 640 [512+128] processes. We remark that it is not possible to get a throughput higher than 16 SYPD, since the speedup does not go beyond about 40.

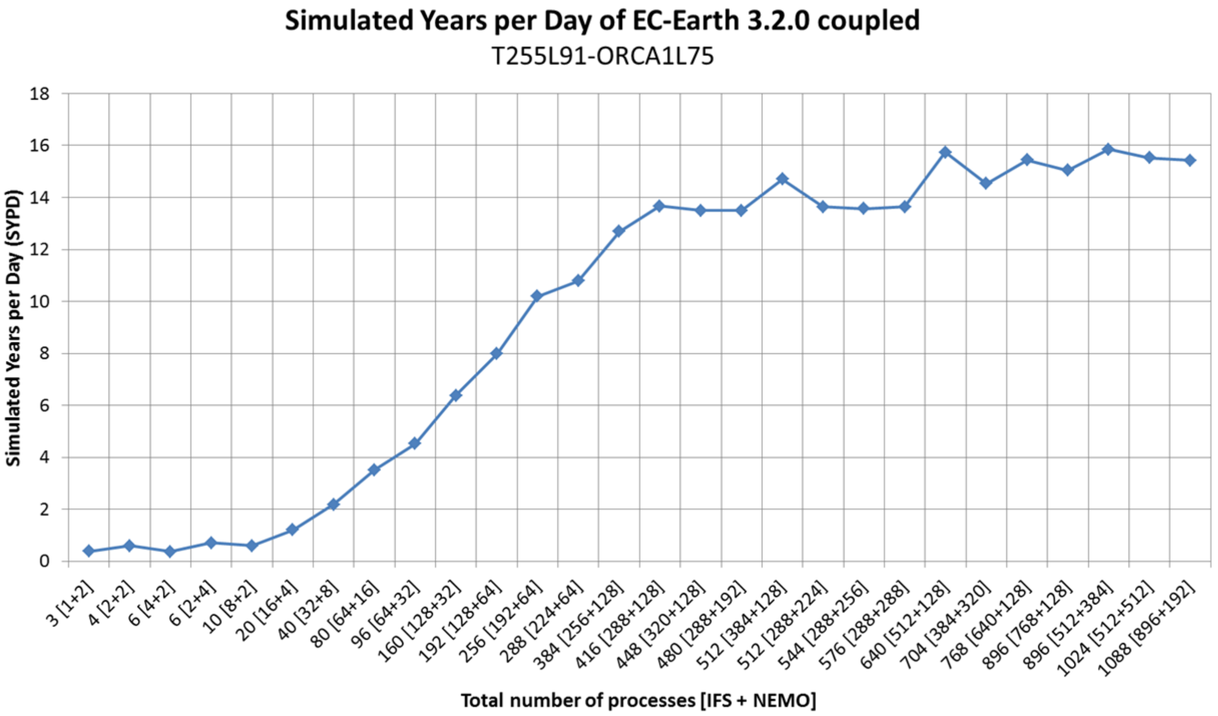

Figure 11: Simulated Years per Day of EC-Earth 3.2.0 coupled in a 2D chart

4.1.3. Performance-efficiency compromise metric

As we explained in section 3, representing 3D data in a 2D chart is quite challenging. To do it, we introduced the metric performance-efficiency compromise which gives a value to each MPI combination. Then applying the technique of selecting the best of each subset, we obtain the mask that we apply later to the common metrics.

Table 4 shows the scores of each MPI combination. There are highlighted the best values in both 1.x (blue) and 2.x (orange) subsets. When the best value coincides in both subsets, it is highlighted in brown. Red and green are used to highlight the worst and the best values respectively.

Perf.-eff. compromise		NEMO processes											
		2	4	8	16	32	64	128	192	224	256	288	320	384	512
IFS processes	1	1	0.7388566	0.39206962											
	2	1.75502733	1.68400603	0.62326996											
	4	0.4374124	0.30922948	0.16308627											
	8	0.69606613	0.63721224	0.45201601											
	16	0.82681647	1.42985835	1.22275921	0.89296316	0.59878015	0.36023554	0.1997723	0.13925388						
	32	0.53793904	1.1179155	2.36018977	2.19958278	1.65679091	1.10452727	0.65564668	0.47184536						
	64				3.0490711	4.17841384	3.05030581	2.03774003	1.51326336	1.34236234	1.03640388	1.09200895	0.88816044	0.86401138	0.64966174
	128				2.24299986	4.98950028	6.53138926	4.85535149	3.86717612	3.46957582	3.17349144	2.95085301	2.70827448	2.339009	1.800255
	192				1.55810548	4.0940055	7.98414277	6.75861176	5.74320734	5.27551738	4.07892876	4.53211619	4.23235051	3.30090493	2.82805647
	224				1.34378147	3.84513106	7.93534449	7.84658642	6.32999784	5.74775644	3.97555562	5.16524418	4.83540256	4.24868756	3.38645377
	256				1.18302556	3.48609135	7.19920256	8.23302911	6.80681537	6.45686996	5.81428035	5.6972382	5.35578537	4.66736746	3.91176847
	288				1.0676774	3.14580445	7.26135314	8.817858	7.46172035	7.13739703	6.64186004	6.34435292	5.75337782	5.11030468	4.11450421
	320						6.46371472	7.99470038	6.86602147	6.75160611	6.39269067	5.99529042	5.70991762	5.05588793	4.24098533
	384						5.69331865	8.29863415	7.34652681	7.00739122	6.35038904	6.08845776	5.8980696	5.2849513	4.40020058
	512						4.36345954	7.59974679	6.92903257	6.5892648	6.09991133	5.88111061	5.83745277	5.50833633	4.62106145
	640						3.70858587	6.09991133	5.88877079	5.98133297	5.10370607	5.32620625	4.68373724	4.21794693	4.45917766
	768						3.1280112	4.96253156	4.66411952	4.88880937	4.77763886	4.53298282	4.46393447	4.16124544	3.68624888
	896						2.38060335	3.91149121	4.29967079	3.89910901	3.78557354	4.09602947	3.49892381	3.46411602	3.63751916

🟥 MIN. SCORE 🟩 MAX. SCORE 🟧 + 🟦 = 🟫

Table 4: Performance-efficiency compromise table with the best values of the subsets 1.x (blue) and 2.x (orange) highlighted. In red and in green are highlighted the worst and the best values respectively.

Finally, if we take all the highlighted combinations and we sort them by ascending order of the total MPI processes, we get the performance-efficiency compromise chart of Figure 12.

From this chart we can draw some points:

- The 416 [288+128] combination has the best performance-efficiency compromise.
- The 6 [4+2] combination has the worst performance-efficiency compromise, which is also the slowest in terms of execution time.
- In section 4.1.1 we pointed out 640 [512+128] and 896[768+128] combinations as the fastest ones, but it was only worth to take into account the 640 [512+128] one. With this chart we can see that we were right, because if we compare their values, clearly the 640 [512+128] combination has better performance-efficiency compromise.

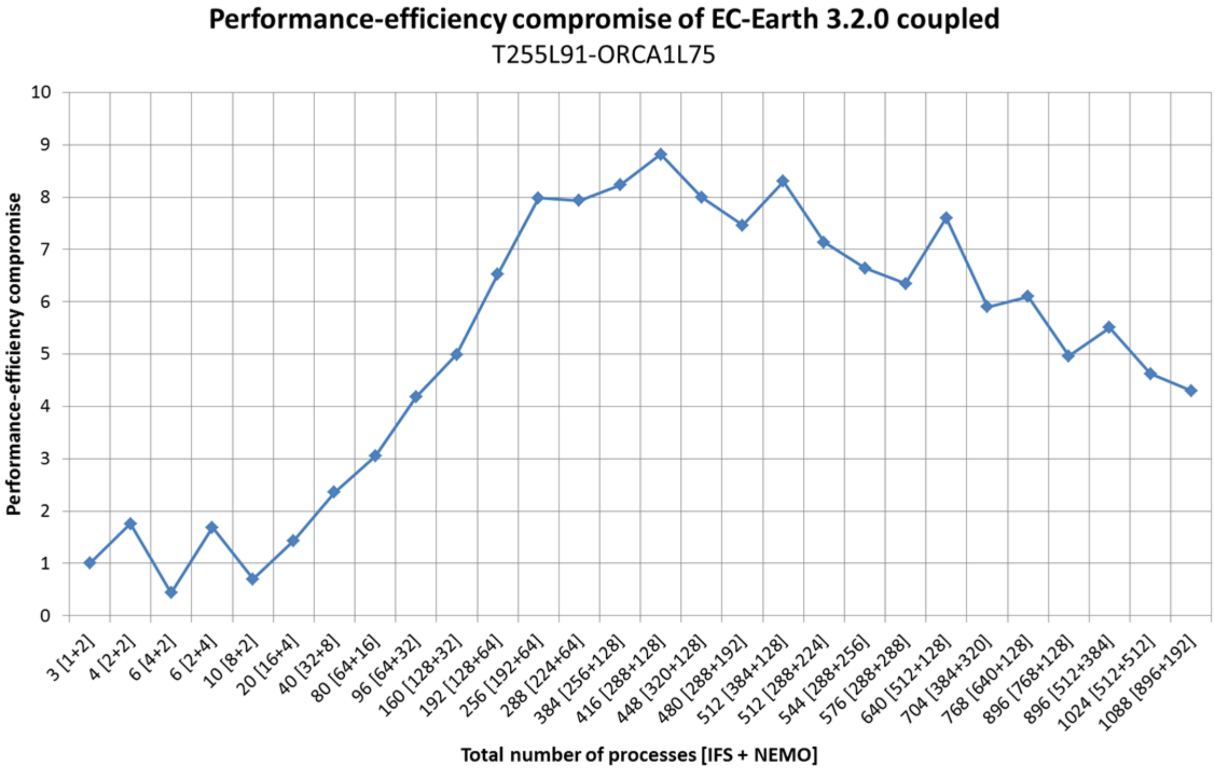

Figure 12: Performance-efficiency compromise of EC-Earth 3.2.0 coupled in a 2D chart

On the other hand, if one is mainly concerned about the efficient usage of resources, this metric reflects that it is better to fix an upper limit, up to 416 [288+128] processes, and also 512 [384+128] and 640 [512+128]. It is important to understand that when we increase the performance (or throughput), we lose efficiency. On the other hand, if we need the simulations to finish in a reasonable time, it is necessary to put a lower limit, around 256 [192+64] processes (also keeping the previous upper limit).

This chart also can be used to compare combinations with the simulated years per day chart.

If an EC-Earth user wants to know the combination that achieves a desired throughput, but there are 2 or more combinations that achieve it, he or she can compare these combinations using the performance-efficiency compromise chart and use the one with the best value. For example, there are 6 combinations that give around 13.6 SYPD: 416 [288+128], 448 [320+128], 480 [288+192], 512 [288+224], 544 [288+256] and 576 [288+288]. But we should choose 416 [288+128] processes because it is the one with the highest performance-efficiency compromise. Obviously, it will be always the combination that uses fewer processes.

5. Conclusions

The execution of EC-Earth needs a lot of resources and they are not unlimited, so it is necessary to use them as efficiently as possible. For this reason, in this study we have presented a scalability analysis in order to know how the release 3.2.0 of EC-Earth scales on MareNostrum III, which are the MPI combinations with the best performance-efficiency compromise and which are the combinations that achieve a desired throughput expressed in SYPD.

To achieve this, it has been necessary to develop a methodology to properly study a bi-dimensional scalability, as we are doing the scalability analysis using two parallel models coupled, IFS and NEMO, and for both we suppose that it is necessary to set different amounts of MPI processes. This arises a problem, which is the representation of 3D data in an understandable way, simplifying the search in order to use 2D data. For doing this, it is necessary to find a way to reduce the amount of combinations, but keeping the most relevant ones. To solve it, we have presented a new metric called performance-efficiency compromise which is calculated as the product of the speedup and the efficiency. In the end, we have got 2D charts that easily show how EC-Earth behaves.

We have seen that EC-Earth does not scale well, as the efficiency is really low. However, we have pointed out which are the most interesting MPI combinations to be used, depending on the needs of the user, i.e., if the performance of the model is more important for a particular study, if it is preferable a good balance between performance and efficiency, etc. In particular, the fastest combination is 640, using 512 MPI processes for IFS and 128 for NEMO. It has a speedup of 40.3 and gives a throughput of 15.7 SYPD. On the other hand, using the new performance-efficiency compromise metric, we see that the best combination is using 416 MPI processes, 288 for IFS and 128 for NEMO. It has a speedup of 35 and is able to achieve about 13.6 SYPD.

Because of there are results from both computational and scientific perspectives, anyone can also make its own conclusions and use the information in its own way.

Acknowledgements

The research leading to these results has received funding from the EU ESiWACE H2020 Framework Programme under grant agreement n° 675191.

References

[1] Asif M., A. Cencerrado, O. Mula-Valls, D. Manubens, A. Cortés and F.J. Doblas-Reyes, 2014: “Case Study in Large Scale Climate Simulations: Optimizing the Speedup/Efficiency Balance in Supercomputing Environments”. 14th International Conference on Computational Science and Its Applications, doi:10.1109/ICCSA.2014.57

[2] Hazeleger W., X. Wang, C. Severijns, S. Ştefănescu, R. Bintanja, A. Sterl, K. Wyser, T. Semmler, S. Yang, B. van den Hurk, T. van Noije, E. van der Linden, K. van der Wiel, 2011: “EC-Earth V2.2: description and validation of a new seamless earth system prediction model”. Clim Dyn, doi:10.1007/s00382-011-1228-5